

CONNECTIONIST MODEL-BASED STEREO VISION FOR TELEROBOTICS

William Hoff and Donald Mathis
Martin Marietta Astronautics Group

SUMMARY

Autonomous stereo vision for range measurement could greatly enhance the performance of telerobotic systems. Stereo vision could be a key component for autonomous object recognition and localization, thus enabling the system to perform low-level tasks, and allowing a human operator to perform a supervisory role. The central difficulty in stereo vision is the ambiguity in matching corresponding points in the left and right images. However, if one has *a priori* knowledge of the characteristics of the objects in the scene, as is often the case in telerobotics, a model-based approach can be taken.

In this paper, we describe how matching ambiguities can be resolved by ensuring that the resulting three-dimensional points are consistent with surface models of the expected objects. A four-layer neural network hierarchy is used in which surface models of increasing complexity are represented in successive layers. These models are represented using a connectionist scheme called *parameter networks*, in which a parametrized object (for example, a planar patch $p=f(h,m_x,m_y)$) is represented by a collection of processing units, each of which corresponds to a distinct combination of parameter values. The activity level of each unit in a parameter network can be thought of as representing the confidence with which the hypothesis represented by that unit is believed. Weights in the network are set so as to implement gradient descent in an energy function.

1. INTRODUCTION

The goal of autonomous stereo vision is to determine the three-dimensional distance, or depth, of points in a scene from a stereo pair of images. This information is very useful for higher level perception capabilities such as autonomous object recognition and localization. Such capabilities could greatly enhance the performance of telerobotic systems, by enabling the system to perform low-level tasks and allowing the human operator to perform a more supervisory role. This paper describes a new approach to stereo vision that we have implemented and with which we have obtained preliminary results. Section 2 provides an overview of stereo vision and our general approach. Section 3 describes our implementation of the approach with a connectionist, or neural network model. Sections 4 and 5 give additional details of the implementation. Section 6 describes results, and Section 7 gives conclusions.

2. STEREO VISION

The usual approach to stereo vision includes the following steps [1]: (1) Features are extracted from each image independently, (2) features from one image are matched with the corresponding features from the other image, thus deriving their depths, and (3) a surface is interpolated between the possibly sparse depth points. The central difficulty in autonomous stereo vision is the second step; *i.e.*, matching corresponding points in the left and right images [2]. The reason is that matching is highly ambiguous, since there is little information to characterize low-level features (*e.g.*, edge points) uniquely. Although less ambiguous higher level features such as long line segments [3] can be used, these approaches are less general and do not explain the ability of the human visual system to fuse stereograms consisting solely of random dots [4].

A different approach to solving the correspondence problem, which we have taken here, is to use a model-based approach, which requires some *a priori* knowledge of the characteristics of the objects in the scene. Specifically, parametrized surface models are defined, such as planar and quadratic patches, and only those matches are allowed that are consistent with those models. The surfaces of the objects in the scene are assumed to be composed of a set of these patches. Low-level features (*i.e.*, edge points) are

used, which makes the approach general in the sense that no specific markings or texture patterns are required. However, the specific surface models restrict the applicability of the approach to situations where the objects in the scene are indeed describable by these surface models. This is a reasonable approximation in most cases, especially for the man-made objects which would be encountered in many situations in telerobotics.

To use this surface-based approach, the processes of matching and surface interpolation must be integrated. This is because matching provides surface depth values at the locations of the matched features, which constrain the interpolated surface. However, the correctness of the choice of matches is judged by the type of surface produced. Therefore, the interpolation process should be integrated with matching so that acceptable matching decisions can be made.

A stereo vision algorithm that integrated matching and surface interpolation was recently developed by one of the authors (Hoff) [5]. Similar approaches have been taken by other researchers [6-8]. The work described in this paper primarily differs from these other approaches in that it uses a connectionist implementation. This approach tightly integrates matching and surface interpolation and naturally combines top-down and bottom-up processing.

3. CONNECTIONIST IMPLEMENTATION OF STEREO CORRESPONDENCE

In connectionist or neural network architectures, a large number of simple processing units are connected by weighted connections. In our stereo implementation, each unit represents a hypothesis about a parametrized model; *i.e.*, a feature correspondence or a surface patch. Each unit has an activation value, which represents the confidence of the hypothesis. Ballard calls this class of models parameter networks [9]. Positive and negative connections between pairs of units represent consistency constraints between the hypotheses they represent. The units then incrementally update their activation values in parallel, based on the activation values of their neighbors and the connections to them. It is this parallel computation which can be used to tightly integrate matching and surface interpolation.

4. DETAILED DESCRIPTION OF THE MODEL: STRUCTURE

4.1 Hierarchical Structure of the Model

Our model consists of a four-layer neural network hierarchy, in which features of increasing complexity are represented in successive layers of the hierarchy. The model takes two stereo images as input at the lowest layer, and produces surface depth and orientation estimates in the highest layer (fig. 1).

Level 1.

The first level of the hierarchy consists of two stereo images that have been preprocessed to detect edge point features. It is assumed that the images are of a densely textured surface, so that there is an abundance of edge features in the images. As far as our model is concerned, the details of the edge-detector are not important. (Note: Unlike other approaches [10], the performance of this algorithm is not tied directly to the spatial scale of the edge detector.)

Level 2.

The second layer consists of representations of correspondences between edge point features in the left and right images. Each correspondence has an associated stereo disparity value, from which the actual 3D position of the point can be calculated. Since we are using local representations, one unit is allocated for each possible depth (or disparity) at each pixel location. (Note: the 3D-points and surface patches are all represented in the coordinate system of the left image.) The range of allowable disparities is a parameter of the model, and is the same at all pixel locations¹.

Level 3.

¹ This is a deficiency of the present model compared to those that allow different disparity estimates at different locations. In the present model, to allow for a wide range of disparities across an image, one must allocate units to account for all possible disparities at every pixel location, regardless of the particular disparity estimate at that location.

The third layer consists of representations of overlapping surface patches of constant depth. Each patch covers a region in 3D space of a fixed diameter (in pixels, not in physical space), and at a constant depth. The representation of the entire surface at this level of the hierarchy would consist of a surface patch at each (x,y) location, each patch being a little plane of zero slope and some known depth. In this level the spatial resolution is reduced so that not every (x,y) pixel location contains an estimate. The diameter of the surface patches is a parameter of the model, and is the same for all patches and at all pixel locations. The range of allowable depths is the same as that of level 2.

Level 4.

The fourth layer consists of representations of surface patches of known depth and constant (but possibly non-zero) slope. These patches cover a larger spatial region than the constant-depth patches, and they also overlap spatially. The representation of the entire surface at this level of the hierarchy would consist of a surface patch at each (x,y) location, each patch being a little plane of some known slope and some known depth. The difference between level 3 and level 4 is that the level 4 patches can have non-zero slope. Again, the spatial resolution is reduced. Three parameters are used to represent a surface patch of non-zero slope: ∂ , m_x and m_y - the depth, and the x and y components of the slope. One unit is used to represent each (∂, m_x, m_y) combination.

Additional Levels.

Additional levels could be defined, such as higher order surface patches (e.g., quadratic surfaces) or volumetric primitives. Higher levels would represent more specialized descriptions and would implement more powerful matching constraints.

4.2 Connections in the Model

Since we have designed the units in the model to represent specific things, we can also design the connections between them. There are two types of connections: (1) Inhibitory, to implement the competition between mutually inconsistent units, and (2) excitatory, to implement the support that mutually compatible units can give to each other. All connections in the model are bi-directional, so the model contains feedback of activation from higher layers to lower layers. Thus, the model can be seen as an instance of an interactive activation and competition model [11].

Inhibitory Connections.

Since separate units are used to represent each possible surface hypothesis at each pixel location, they represent *conflicting estimates* or *inconsistent hypotheses* of the surface at that point. We would therefore like these units to compete against each other, and so at each level, the units located at the same (x,y) position are fully connected with inhibitory connections. In level 4, this means that each of these 3D grids of units forms a giant competing pool of units - there can be only one winning (∂, m_x, m_y) combination at a given patch location.

Excitatory Connections.

To set excitatory connections, units are connected to other units that support each other. Each level 2 unit represents a 3D-point with a particular disparity (∂) at a particular (x,y) location. As shown in figure 2, it receives input from level 1 from the two locations that it accounts for — one at (x,y) in the left image, and one at $(x+\partial,y)$ in the right image.

Each level 3 unit represents a surface patch at a particular constant depth ∂ and location (x,y) , and receives input from all 3D-point units (in level 2) in a local region that lie sufficiently close to the (x,y) location and (∂) depth represented by the level-3 unit (see fig. 3). Each level 4 unit represents a surface patch at a particular depth ∂ , location (x,y) , and slope (m_x, m_y) and receives input from all level-3 units that lie sufficiently close to the (x,y) location, (∂) depth, and (m_x, m_y) slope represented by the level-4 unit (see fig. 4).

The size of the support regions imply a characteristic scale assumption in the model. The support widths are parameters of the model. For example, a *support-region width* of w pixels from level 2 units to level 3 units embodies the assumption that the surface is smooth at the scale of w pixels. The disparity range

and surface slope range indicate assumptions about the allowable ranges for distance and orientation of the visible surfaces.

5. DETAILED DESCRIPTION OF THE MODEL: WEIGHTS

Now that we have the structure of the model — *i.e.*, the layers, units, and connections are defined — we must assign weights to the connections. This is not a trivial problem, since it is difficult to specify in advance the relative importance of the various components, yet the performance of the model depends critically on setting the weights properly. One possibility is to have the network learn the weights on its own. This introduces additional problems, however, although we would like to explore this approach eventually. For example, one must create a training set of data that covers the important examples, and in the appropriate proportions. Also, training a network with feedback is computationally expensive [12].

Instead, we have chosen to set the weights so as to implement the minimization of an energy (or "cost") function. Specifically, the weights implement gradient descent in this energy function [13]. This increases the likelihood of the system settling into a stable state, and a state that satisfies the requirements we wish to impose on the system (e.g., winner-take-all behavior, etc). The use of an energy function also greatly increases our ability to analyze the model and make meaningful adjustments to its parameters.

The basic idea is that an energy (or "cost") function is defined as a function of the activations of all the units in the network, and this energy function is designed to be minimized when the pattern of activity in the network corresponds to a good solution to the problem the network is trying to solve. Each unit then locally computes the gradient of the energy function with respect to its own activation, and adjusts its activation in the direction that reduces the energy. This process is repeated until the network settles on a stable pattern of activity.

There are many variations of gradient descent, including the IAC model, the BSB model [14], and Grossberg's model [15]. We used a very simple activation function:

$$a_j(t+1) = a_j(t) + \Delta a_j(t), \quad (1)$$

$$\begin{array}{ll} \text{where} & \Delta a_i(t) = -\epsilon * \text{grad}_i * (1 - a_i(t)) \quad \text{if } \text{grad}_i \leq 0 \\ \text{or} & \Delta a_i(t) = -\epsilon * \text{grad}_i * a_i(t) \quad \text{if } \text{grad}_i > 0 \end{array}$$

Here, grad_i is the rate of change of energy with respect to a_i , or $\partial E / \partial a_i$. This is based on gradient descent, but is not exactly the same as gradient descent since the activations (a_i) are bounded by 0 and 1.

5.1 The Energy Function

There are many different energy functions that can be used to implement our model. Some energy functions contain more terms than others, and some seem to be easier to understand than others. We made the decision to try to minimize complexity, while still using a function that would provide good results. The energy function that was used is a sum of six terms. Each term represents a constraint on good solutions to the surface estimation problem:

$$E(\mathbf{a}) = \quad (2)$$

$$\begin{aligned} & - \sum_{i \in L2} e_i a_i && \text{(Term 1: Image Evidence)} \\ & + k1 \sum_{c \in L2} \sum_{i \in c} \sum_{j \in c, j \neq i} a_i a_j && \text{(Term 2: WTA - Level 2)} \end{aligned}$$

$$- k_2 \sum_{i \in L_2} \sum_{j \in S_i^{L_3}} a_i a_j f_{23}(i,j) \quad (\text{Term 3: } L_2 \Leftrightarrow L_3 \text{ support})$$

$$+ k_3 \sum_{c \in L_3} \sum_{i \in c} \sum_{j \in c, j \neq i} a_i a_j \quad (\text{Term 4: WTA - Level 3})$$

$$+ k_4 \sum_{c \in L_4} \sum_{i \in c} \sum_{j \in c, j \neq i} a_i a_j \quad (\text{Term 5: WTA - Level 4})$$

$$- k_5 \sum_{i \in L_4} \sum_{j \in S_i^{L_3}} a_i a_j f_{34}(j,i) \quad (\text{Term 6: } L_3 \Leftrightarrow L_4 \text{ support})$$

where the function $f(i,j)$ is a "closeness" function used to weight the support of a point i for a plane j within a threshold distance "d" by an amount inversely proportional to the point's distance from the plane:

$$f(i,j) = 0 \quad \text{if distance}(\text{point}(i), \text{plane}(j)) > d \quad (3)$$

$$\text{or} \quad f(i,j) = \frac{d - \text{distance}(\text{point}(i), \text{plane}(j))}{d} \quad \text{otherwise.}$$

The energy function (Eq. 2) has six terms, but there are actually only three different types of terms used. The first type of term (term 1) is minimized when the level 2 units maximally "agree" with the image evidence. The second type of term is minimized when winner-take-all situations exist wherever they are desired. There are three terms of this type (terms 2,4,5), one for each layer in which WTA situations are desired (the "c" parameter ranges over all competitive pools in the layer). The third type of term is minimized when hypotheses in one layer maximally "agree" with related hypotheses in adjacent layers (terms 3,6). The "S" variables are the units' "support regions" (e.g., $S_i^{L_3}$ is the set of units in Level 3 that support unit i).

The constants k_j are used to represent the relative importance of the various terms within the overall energy function. By inspection, it can be seen that the WTA terms are minimized when no more than one unit has a nonzero activation. The "support" terms are all functions of a single unit in one layer and a group of units in its "support region" in an adjacent layer. These terms are minimized when the unit with the most support within a competitive pool is the winner in that pool. The "Image Evidence" term (term 1) is minimized when the 3D point unit with the most support within a competitive pool is the winner in that pool.

5.2 Connection Weights

The connection weights in the network are derived from the energy function by taking derivatives with respect to the unit activations. In order for a unit to be able to calculate the derivative locally, it usually needs to know the activations of some other set of units. This communication is implemented with weighted connections between the units. For units in level 2 of the present model, this derivative takes the form:

$$\text{grad}_i = \frac{\partial E}{\partial a_i} = k_1 \sum_{j \in c_i} a_j - (k_2 + k_3) \sum_{k \in S_i^{L_3}} a_k f_{23}(i,k) - e_i \quad (\text{with } j \neq i) \quad (4)$$

where c_i is a "competitive pool" of units of which unit i is a part, and $S_i^{L_3}$ is the "support region" in Level 3 for unit i .

The unit i can compute this derivative if we create connections between it and all the other units that appear in the above equation. The unit can then calculate the gradient using a net input function such as:

$$\text{net}_i = \sum_j w_{ij} a_j - e_i \quad (5)$$

where the subscript j ranges over all of the units to which unit i is connected. The term e_i is the "external input" to the unit, which in this case is the input from the edge images. To make this technique work, however, the weights (the w_{ij} 's) must be set to the following values:

$$\begin{aligned} w_{ij} &= k_1, & \text{for all units } j \in c_i, \text{ with } j \neq i. \\ w_{ik} &= -k_2 f_{23}(i,j), & \text{for all units } k \in S_i^{L3}. \end{aligned} \quad (6)$$

To put it another way, the connectivity and the connection weights themselves are derived from the energy function by first deriving the equation for $\partial E / \partial a_i$ from the energy function, and then choosing a set of connections and a net input function that *implement* the $\partial E / \partial a_i$ equation. This technique was used to determine the connectivity and weights throughout the network.

Note that the energy function still contains 5 unknown parameters (the k_i), which must be selected by trial and error. However, we can derive some useful relationships between these parameters, by strictly enforcing the winner-take-all requirement. For example, suppose that C is a set of units that forms a competing pool (i.e., a "winner-take-all" pool). If the current pattern of activations of the units in C is a winner-take-all pattern (i.e., there is one unit with a "1" activation, and all of the others have a "0" activation), then this activation pattern will *remain* stable if:

$$\begin{aligned} \text{grad}_w &\leq 0 & \text{for the "winning" unit } (w \in C), \text{ and} \\ \text{grad}_l &\geq 0 & \text{for all the "losing" units } (l \in C). \end{aligned} \quad (7)$$

This is true because in gradient *descent*, the activation values are changed in a direction opposite in sign from that of the gradient, so if the above conditions hold, the winning unit will try to *increase* its activation (but will be limited at 1), and the losing units will try to *decrease* their activations (but will be limited at 0). Therefore the winner-take-all pattern is stable¹.

When these conditions are applied to the competing pools in our model, we get the following results:

$$k_1 \geq (k_2)W_{2,3} + e_i \quad (8a)$$

$$k_3 \geq (k_2)W_{2,3} + (k_5)W_{3,4} \quad (8b)$$

$$k_4 \geq (k_5)W_{3,4} \quad (8c)$$

where $W_{a,b}$ is the width of the support region between units in Level a and units in Level b . So given values for k_2 and k_5 , we can set the other k values according to these equations and be sure that all winner-take-all patterns will be stable.

6. RESULTS

The system was run primarily on 1-dimensional random-dot stereograms. (A 1-d stereogram image is just a single horizontal row of binary pixels). The system was run many times with different parameter settings to study the effects of enforcing WTA behavior to varying degrees. It was found that if the k parameters were set to values that would strictly enforce WTA, the network settled on *very poor* solutions to the surface estimation problem. Better results were achieved when the parameters governing the lower levels were set much lower than that required to enforce WTA (i.e., less than half the value), and the higher-level parameters were set just slightly lower than that required for WTA. The best results were achieved

¹ Note that this does not ensure that the network will *settle* on a winner-take-all pattern, but only that *if* a winner-take-all pattern exists, it will remain stable.

when, instead of setting the parameters to fixed values, they were all set initially to values far below the WTA values, and then slowly increased to the WTA values. This appeared to work better because it allowed partial activation of large numbers of "almost right" hypotheses initially, and then as the parameters were raised, the best hypotheses were "selected" (more or less) from the set of partially active hypotheses.

How to interpret the graphics:

Each diagram consists of three large rectangles filled with small squares. The highest rectangle represents the 3d-point layer of the network, the next lower rectangle represents the constant-depth-patch layer, and the lowest rectangle represents the constant-slope layer. Each rectangle is a graphic representation of the activations of the units in that layer of the network.

The 3d-point layer of the net is a 2D parameter space, (x vs. *disparity*). The horizontal dimension is x , (it ranges from $x=0$ to $x=50$), and the vertical dimension is *disparity* (-5 to $+5$). Each point in the space contains a unit, and the size of the black square at that point represents the activity of the unit. The constant-depth layer also has these dimensions. The constant-slope layer of the network has an extra dimension: *slope*. This extra dimension is graphically displayed as another level of rectangles-within-rectangles. The large rectangle contains a horizontal row of "tall" rectangles. There is one of these for every other x value (the spatial resolution was reduced by a factor of 2). Each of these tall rectangles is filled with 77 tiny rectangles, each of which represents a distinct combination of (x , *disparity*, and *slope*). These tiny rectangles are arranged in a 2D grid, the vertical dimension of which is *disparity* (just as it is in the lower layers), and the horizontal dimension is *slope*. The slope ranges from -0.6 to $+0.6$, in increments of $.2$.

Figures 5a-5c depict the history of one run of the network after 1, 5 and 9 iterations. The input to the network is a stereo pair of images of a surface with slope -0.2 (this would appear as a diagonal line running from the top-left to the bottom-right of the large rectangles. Within the large "tall" rectangles, a slope of -0.2 would appear as a filled-in tiny rectangle three from the right (slope increases right to left; zero slope is in the center). By the end of the series, you can see that the disparity has been estimated fairly well. The slopes are roughly correct throughout the image ($\pm .2$), and they are exactly correct at 15 of the 25 pixels, or in 60% of the image).

7. CONCLUSIONS

The system performed very well on the 1D stereogram data. Addition of the second spatial dimension will require an extension of the support regions into the y dimension, but will not require a change to the competitive pools, since they extend only in the x direction. The energy function itself will not change (except for the fact that the S_i 's will change). The changes will be primarily *quantitative*, not *qualitative*. An improvement in performance is expected, however, since the extension of the "cooperative" constraints (smoothness, planarity) into the y dimension will allow 3d-point hypotheses to take into account the smoothness (or lack thereof) of the surface in the y dimension. There will be an impact on the derivations of the k -values needed to preserve WTA, since there will be more competitive pools in the support region of a given hypothesis.

It is still not known exactly how the values of the k coefficients should be changed over time to maximize performance. But since the network is trying to find an energy minimum (ideally, the *global* minimum), simulated annealing with a fixed set of k 's would probably produce equal or (more likely) *better* results than the "pseudo-annealing" that was used here.

Despite the attempts to derive the values of as many of the model's parameters as possible, some amount of "rapid prototyping" was still ultimately necessary. For example, there was little guidance in picking the relative values of the "support weight" parameters (k_2 and k_5). For our purposes, we made them equal.

REFERENCES

- [1] Barnard, S. and M. Fischler, "Computational Stereo," *ACM Computing Surveys*, Vol. 14, No. 4, Dec 1982, pp. 195-210.
- [2] Marr, D., *Vision*, Freeman, San Francisco, 1982.
- [3] Medioni, G. and R. Nevatia, "Segment-based Stereo Matching," *Computer Vision, Graphics, Image Processing*, Vol. 31, July 1985, pp. 2-18.
- [4] Julesz, B., *Foundations of Cyclopean Perception*, University of Chicago Press, Chicago, 1971.
- [5] Hoff, W. and N. Ahuja, "Surfaces from Stereo: Integrating Feature Matching, Disparity Estimation, and Contour Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 11, No. 2, February 1989, pp. 121-136.
- [6] Olsen, S., "Concurrent Solution of the Stereo Correspondence Problem and the Surface Reconstruction Problem," *Proc. Eighth Int. Conf. Pattern Recognition*, 1986, pp. 1038-1041.
- [7] Eastman, R. and A. Waxman, "Using Disparity Functionals for Stereo Correspondence and Surface Reconstruction," *Computer Vision, Graphics, Image Processing*, Vol. 39, 1987, pp. 73-101.
- [8] Boulton, T. and L. Chen, "Synergistic Smooth Surface Stereo," *Proc. IEEE Second Int. Conf. Computer Vision*, Dec 1988, pp. 118-122.
- [9] Ballard, D., "Parameter Nets," *Artificial Intelligence* 22, 1984, pp. 235-267.
- [10] Marr, D. and T. Poggio, "A Theory of Human Stereo Vision," *Proc. Royal Soc. London*, Vol. B204, pp. 301-328, 1979.
- [11] McClelland, J. and D. Rumelhart, "An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of Basic Findings," *Psychological Review*, 1988.
- [12] Hinton, G. and T. Sejnowski, "Learning and Relearning in Boltzmann Machines," D. Rumelhart and J. McClelland, in *Parallel Distributed Processing*, Vol. 1, Ch 7., MIT Press, 1986.
- [13] Hopfield, J. and D. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, Vol. 52, 1985, pp. 141-152.
- [14] Anderson, J., "Neural Models with Cognitive Implications," in D. LaBerge and S. Samuels (Eds.), *Basic Processes in Reading Perception and Comprehension*, pp. 27-90, Hillsdale, NJ: Erlbaum, 1977.
- [15] Grossberg, S., "A Theory of Visual Coding, Memory, and Development," E. Leeuwenberg and H. Buffart (Eds.), *Formal Theories of Visual Perception*, New York: Wiley, 1978.

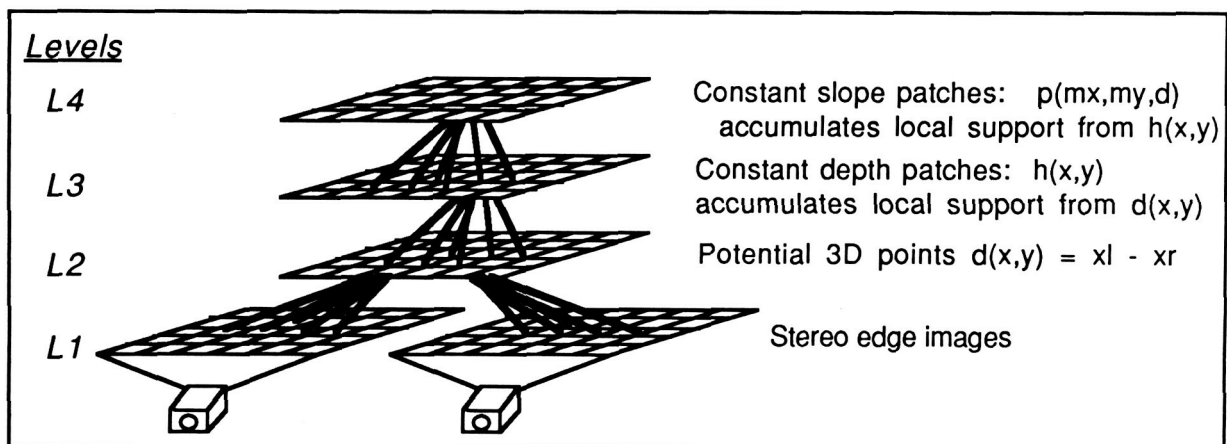


Figure 1. Four level hierarchical model for stereo matching and surface representation.

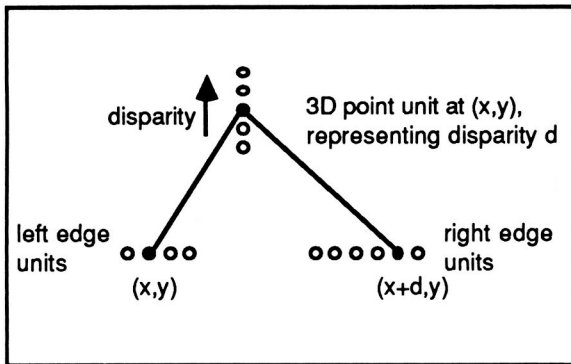


Figure 2. A 3D point unit receives support from left and right edge point units.

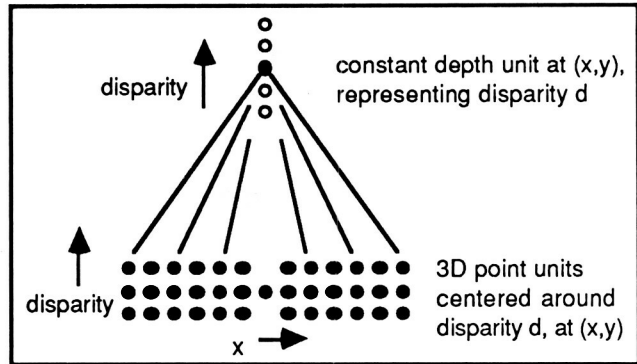


Figure 3. A constant-depth patch unit receives support from 3D point units.

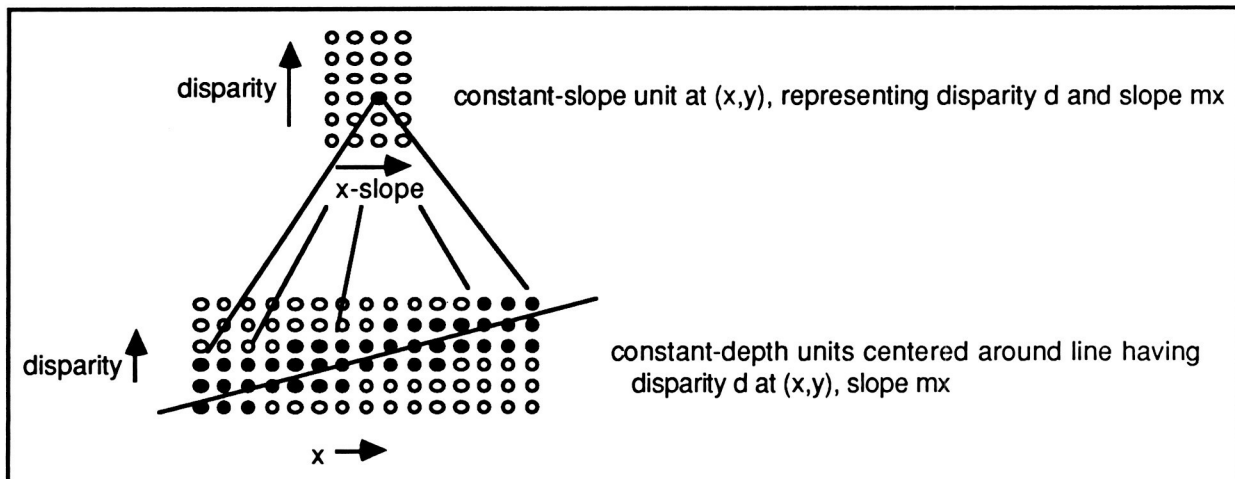


Figure 4. A constant-slope patch unit receives support from constant-depth units.

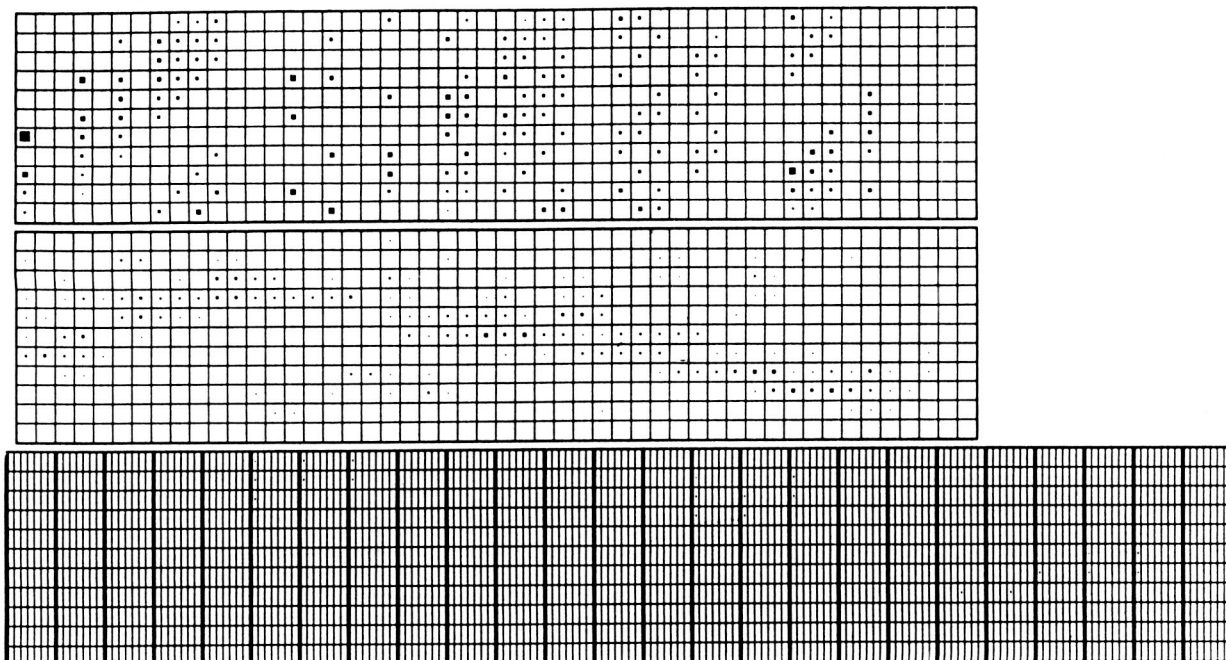


Figure 5a. The network after 1 iteration.

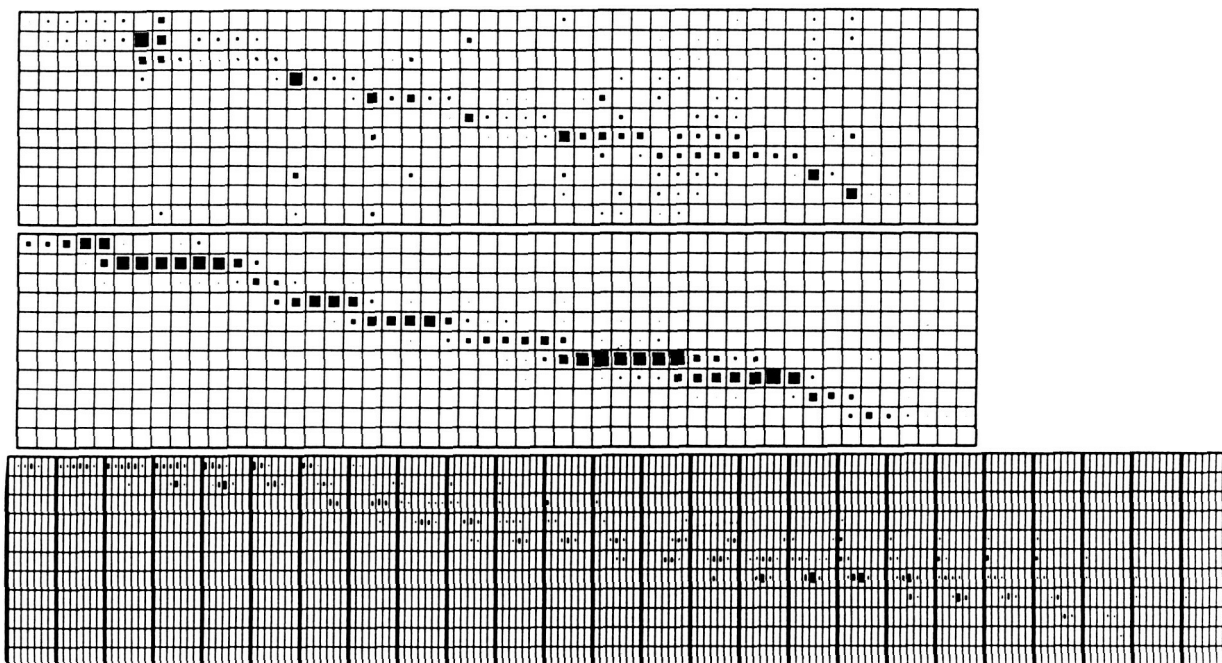


Figure 5b. The network after 5 iterations.

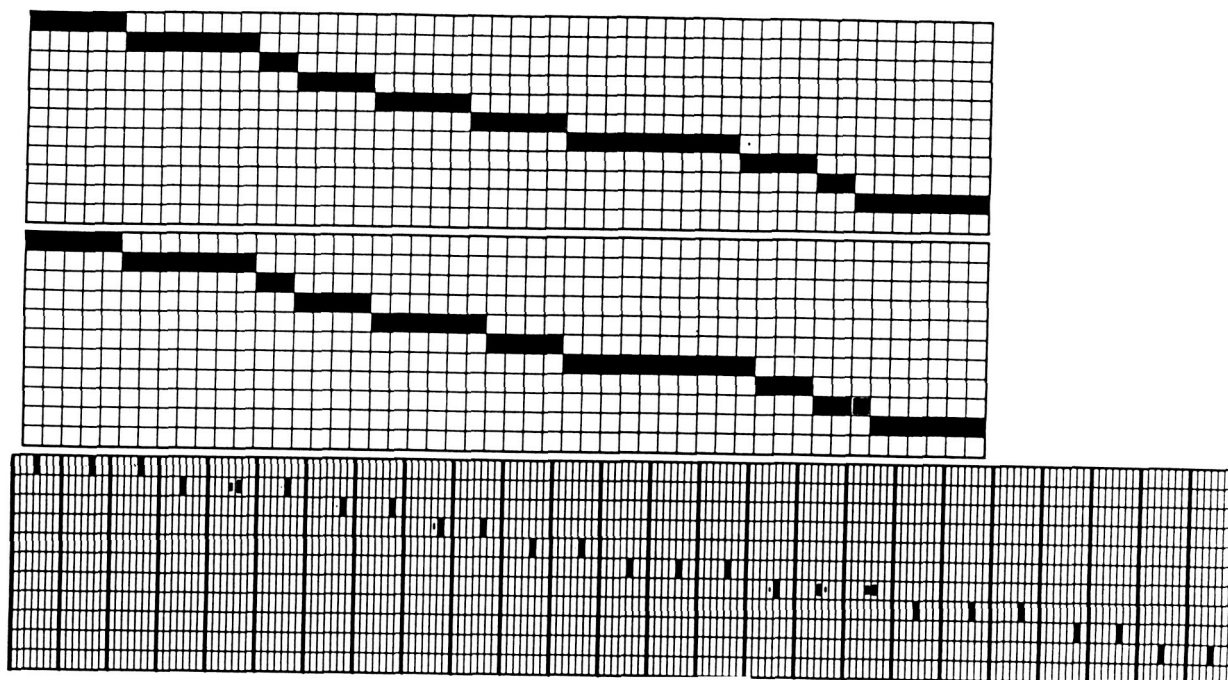


Figure 5c. The network after 9 iterations.